II. AMENDMENTS TO THE SPECIFICATION:

Please make the following amendments to the specification:

In the Abstract, please amend as follows:

Under ~~then~~ the present invention, a native application is packaged within an <u>Open Service</u> <u>Gateway Initiative (OSGi)</u> bundle to create a link there between. Then, the OSGi bundle is installed within an OSGi environment of a client device. Once installed, the OSGi bundle is deployed in a native environment of the client device and the native application is removed from within the OSGi bundle while maintaining the link. Thereafter, the native application within the native environment can be controlled from the server using the OSGi bundle within the OSGi environment.

On page 1, para. 0002, please amend as follows:

As computer networking has become more advanced, a standard known as the Open Service Gateway Initiative (OSGi) has been developed <u>(OSGi is a trademark, registered trademark, or</u> <u>service mark of The OSGi Alliance in the US and other countries)</u>. The OSGi is an industry plan to provide a standard way to deliver managed services to devices and local networks. With such a standard, home users could, for example, change the setting on their thermostat from a remote location (e.g., the workplace). In general, the OSGi provides a good framework for developing application components. Under the OSGi, a basic component is known as an OSGi bundle. An OSGi application can be made up of combinations/suites of bundles that might use common functionality. To this extent, the OSGi allows developers to define the dependencies between the bundles such as the packages and services required by the bundles. The OSGi runtime can also determine whether a device has the necessary packages and resources. In a typical

implementation, an OSGi architecture will include, among other components, a server and one or more client devices. Each client device will have an OSGi environment within which OSGi applications are deployed. Using a management program on the server, the functions of the OSGi applications can be controlled. One type of function often performed on OSGi client devices is life cycle management. Life cycle management allows OSGi applications to be started, stopped, updated, installed or uninstalled from the server.

On pages 2-3, para. 0005, please amend as follows:

[0003] Unfortunately, as convenient as the OSGi framework can be, it fails to provide for similar management/control of native applications on the client device. For example, a standard desktop (e.g., WIN-32 {WIN32 is a trademark, registered trademark, or service mark of Microsoft Corporation in the US and other countries}) computer could have several native application that run within a native environment. Typical examples of native applications include word processing programs, spreadsheets, etc. Since such applications are widely used, it would be highly advantageous to be able to control them in a similar manner from the OSGi environment. Currently, the only way to control the functions of the native application from the server is to write separate programs for each desired function. Not only is this extremely tedious, but it is also highly inefficient.

On pages 2-3, para. 0005, please amend as follows:

In general, the present invention provides a method, system and program product for controlling (e.g., managing a life cycle of) native applications using OSGi bundles. Specifically, under ~~then~~ the present invention, a native application is packaged within an OSGi bundle to create a link therebetween. Information describing the commands needed to control the life cycle of the

bundle (install, uninstall, start, stop) may also be stored within the OSGi bundle during this packaging step. In any event, the packaged OSGi bundle is installed within an OSGi environment of a client device. Once installed, the packaged OSGi bundle is deployed in a native environment of the client device and the native application is optionally removed from within the packaged OSGi bundle while maintaining the link. Thereafter, the native application within the native environment can be controlled from the server using the OSGi bundle within the OSGi environment.